

# PATENT ABSTRACTS OF JAPAN

(11)Publication number : 2000-132531

(43)Date of publication of application : 12.05.2000

(51)Int.Cl.

G06F 15/177

G06F 15/16

(21)Application number : 10-301948

(71)Applicant : PFU LTD

(22)Date of filing : 23.10.1998

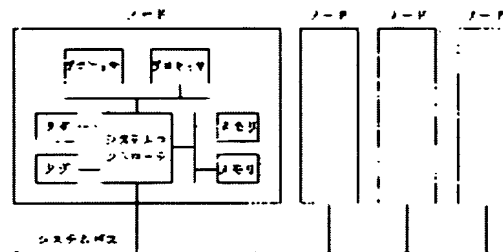
(72)Inventor : HIKONO ATSUSHI  
TAKEBE KENJI  
SATO MASAMI

## (54) MULTIPROCESSOR

### (57)Abstract:

**PROBLEM TO BE SOLVED:** To make effectively performable a process to keep the coherency of caches in a multiprocessor system which includes plural processors having the caches.

**SOLUTION:** When a system controller receives a memory access request from its follower processor, the controller broadcasts the received access request to a system bus and also outputs the cache state included in the tag information corresponding to a memory space x to be accessed to the system bus. On receiving the broadcast memory access request, the system controller of another node outputs the cache state included in the tag information corresponding to the space x to the system bus, gives a request to its follower processor or memory for a necessary process based on the state of the cache existing on the system bus and returns a response to the system controller of a master after the requested process is over.



## LEGAL STATUS

[Date of request for examination]

[Date of sending the examiner's decision of rejection]

[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration]

[Date of final disposal for application]

[Patent number]

[Date of registration]

[Number of appeal against examiner's decision of

rejection]

[Date of requesting appeal against examiner's  
decision of rejection]

[Date of extinction of right]

Copyright (C); 1998,2003 Japan Patent Office

(19) 日本国特許庁 (J P)

(12) 公開特許公報 (A)

(11) 特許出願公開番号

特開2000-132531

(P2000-132531A)

(43) 公開日 平成12年5月12日 (2000.5.12)

(51) Int.Cl.

G 0 6 F 15/177

15/16

識別記号

6 8 2

6 7 4

6 4 5

F I

G 0 6 F 15/177

15/16

特マコト (参考)

6 8 2 J

6 7 4 A

6 4 5

5 B 0 4 5

審査請求 未請求 請求項の数 5 O L (全 16 頁)

(21) 出願番号

特願平10-301948

(22) 出願日

平成10年10月23日 (1998. 10. 23)

(71) 出願人 000136136

株式会社ピーエフユー

石川県河北郡宇ノ気町宇野気ヌ98番地の  
2

(72) 発明者 彦野 厚志

石川県河北郡宇ノ気町宇野気ヌ98番地の  
2 株式会社ピーエフユー内

(72) 発明者 武部 建治

石川県河北郡宇ノ気町宇野気ヌ98番地の  
2 株式会社ピーエフユー内

(74) 代理人 100080894

弁理士 京谷 四郎 (外1名)

最終頁に続く

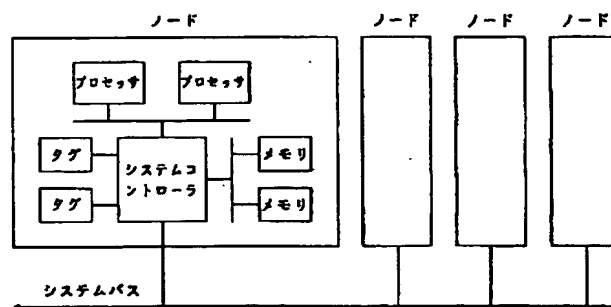
(54) 【発明の名称】 マルチプロセッサ

(57) 【要約】

【課題】 キャッシュを持つ複数のプロセッサを有するマルチプロセッサにおいて、キャッシュのコヒーレンシーを維持するための処理を効率良く行うこと。

【解決手段】 システム・コントローラが配下のプロセッサからメモリ・アクセス要求を受け取ると、当該メモリ・アクセス要求をシステム・バス上にブロードキャストすると共に、アクセス対象のメモリ空間xに対応するタグ情報が持つキャッシュ状態をシステム・バス上に出力する。ブロードキャストされたメモリ・アクセス要求を受け取った他のノードのシステム・コントローラは、メモリ空間xに対応するタグ情報が持つキャッシュ状態をシステム・バス上に出力し、次いでシステム・バス上のキャッシュの状態を参照して配下のプロセッサ又はメモリに対して必要な処理を要求し、要求した処理が終了後にマスタのシステム・コントローラに対して応答を返す。

本発明の原理説明図



## 【特許請求の範囲】

【請求項 1】 キャッシュを持つプロセッサ、システム・コントローラおよび主メモリの一部を構成するメモリを有する複数のノード、並びに複数のノードを接続するシステム・バスを具備するマルチプロセッサにおいて、

各システム・コントローラは、配下のプロセッサからメモリ・アクセス要求を受けた時に、当該メモリ・アクセス要求をシステム・バスを介して他の全てのシステム・コントローラにブロードキャストし、

配下のプロセッサ又は他のシステム・コントローラからのメモリ・アクセス要求を受けた時に、自己のキャッシュのタグ部の中に当該メモリ・アクセスの対象となるメモリ空間に対応するタグ情報が存在するか否かを調べ、存在する場合には該当するタグ情報が持つキャッシュ状態をシステム・バスに出力し、存在しない場合には無効を示すキャッシュ状態をシステム・バスに出力し、システム・バス上のキャッシュ状態を参照し、配下のプロセッサ又はメモリを制御する必要がある場合には配下のプロセッサ又はメモリを制御することを特徴とするマルチプロセッサ。

【請求項 2】 ブロードキャストされたメモリ・アクセス要求を受け取ったシステム・コントローラは、システム・バス上のキャッシュ状態を参照して独自に配下のプロセッサに対する処理が必要か否かを判断し、必要と判断した場合には配下のプロセッサに必要とされる処理を要求することを特徴とする請求項 1 のマルチプロセッサ。

【請求項 3】 自己の配下に存在する複数のプロセッサに対してインバリデート、コピーバック又はコピーバック・インバリデートの処理要求を発行したシステム・コントローラは、上記複数の処理要求に対する処理終了の応答を全て受け取った段階で、メモリ・アクセス要求をブロードキャストしたシステム・コントローラに対して纏めて処理終了の応答を返すことを特徴とする請求項 2 のマルチプロセッサ。

【請求項 4】 メモリ・アクセスの対象となるメモリ空間を持つノードのシステム・コントローラは、メモリ・アクセス要求を受け取った時に自己のメモリに対するアクセスを開始し、次いでシステム・バス上のキャッシュ状態を参照してメモリ・アクセスが必要か否かを調べ、必要ないと判断した場合には当該メモリ・アクセスをキャンセルすることを特徴とする請求項 1 のマルチプロセッサ。

【請求項 5】 メモリ・アクセス要求をシステム・バスにブロードキャストしたシステム・コントローラは、システム・バス上のキャッシュ状態を参照して、プロセッサへの処理が必要なシステム・コントローラ又はリソース・メモリを制御するシステム・コントローラからのみ応答を待ち、関係のないシステム・コントローラからは

応答を待たないことを特徴とする請求項 1、請求項 2、請求項 3 または請求項 4 マルチプロセッサ。

## 【発明の詳細な説明】

【0001】

【発明の属する技術分野】 本発明は、複数のプロセッサのキャッシュのコヒーレンシーの制御を複数のシステム・コントローラがシステム・バスを介して行うマルチプロセッサに関するものである。

【0002】

10 【従来の技術】 従来、複数のプロセッサを制御するシステム・コントローラを複数個持つマルチプロセッサにおいては、プロセッサ又はバス・ブリッジからメモリ・アクセス要求を受けたシステム・コントローラ（マスタのシステム・コントローラ）が他のプロセッサを制御しているシステム・コントローラ（スレーブのシステム・コントローラ）から各プロセッサのキャッシュの状態を読み出し、必要に応じて他のシステム・コントローラが制御するプロセッサが持つキャッシュに対するコピーバック要求やインバリデート要求を当該システム・コントローラに対してシステム・バスを経由して送っていた。

20 【0003】

【発明が解決しようとする課題】 従来のシステムでは、キャッシュのコヒーレンシー（coherency）を保つために、或る一つのメモリ・アクセス要求に対して複数のバス・トランザクションのやり取りがシステム・バス上で必要であった。例えば、或るメモリ・アクセスに対して複数のプロセッサのキャッシュのインバリデート（invalidate）作業が必要な場合は、その数だけメモリ・アクセス要求を受けたマスタのシステム・コントローラがシステム・バスを介してインバリデート作業が必要なプロセッサを配下に持つシステム・コントローラに対してインバリデート要求を送らなければならなかった。

30 【0004】 そのため、幾つものバス・トランザクションがシステム・バスに出てしまい、システム・バスを占有してしまっていた。そのため、プロセッサのキャッシュに対するインバリデート要求やコピーバック要求が遅れてしまうと共に、他のアクセスと競合する確率が高くなり、システム全体の性能の低下を招いていた。本発明は、この点に鑑みて創作されたものであって、複数のプロセッサの各々がキャッシュを有するマルチプロセッサにおいて、キャッシュのコヒーレンシーを維持するための処理を効率よく行い得るようにすることを目的としている。

40 【0005】

【課題を解決するための手段】 図 1 は本発明の原理説明図である。本発明では、キャッシュの状態を複数のシステム・コントローラで共有する。或るメモリ・アクセス要求が発生したら、当該メモリ・アクセス要求を受けたマスタのシステム・コントローラは、システム・バスに

当該アクセス要求をブロードキャストする。システム・バスに接続されている各システム・コントローラは、当該ブロードキャストを受信すると、自分が制御するプロセッサのキャッシュの状態をシステム・バス上に出力する。各システム・コントローラは、システム・バス上のキャッシュの状態を見ることで、配下のプロセッサに対して、コピーバック作業やインバリデート作業が必要か否かを独自に判断することが出来る。

【0006】システム・コントローラは、メモリ・アクセス要求を受け取ると、マスタのシステム・コントローラから要求されることなく、必要に応じてコピーバック作業やインバリデート作業を即座に開始する。

【0007】また、ブロードキャストされたメモリ・アクセス要求がリードの場合には、リソース・メモリを持つシステム・コントローラは、そのメモリへのリード・アクセスを開始し、次いで各システム・コントローラから出力されたキャッシュの状態が最新データがキャッシュに存在することを示しているか否かを調べ、最新データがキャッシュに存在する場合には当該メモリ・アクセスをキャンセルする。なお、リソース・メモリとは、ブロードキャストされたメモリ・アクセス要求の対象となるメモリ空間を持つメモリを意味している。

【0008】処理が必要と判断したシステム・コントローラは、自分が制御するプロセッサやメモリへの処理が全て終了した段階で、マスタのシステム・コントローラに応答を返す。マスタのシステム・コントローラは、応答が必要なシステム・コントローラからの応答を待つと共に、自分が制御するプロセッサへの処理が必要な場合はその処理を行った後、メモリ・アクセスを終結させる。

【0009】本発明によれば、システム・コントローラの各々が独自にキャッシュのコヒーレンシーを維持するための作業を行うために、システム・バスの使用を最低限にすることができ、性能の低下を防げる。また、各々のシステム・コントローラが独自に判断するために、いちはやくプロセッサに対するコピーバック作業やインバリデート作業、メモリ・アクセス、メモリ・アクセスのキャンセルの開始が可能である。

【0010】

【発明の実施の形態】図2は本発明のマルチプロセッサの構成例を示す図である。同図において、100はノード、111と112はプロセッサ、121と122はタグ部、130はシステム・コントローラ、140はクロスバ・スイッチ、151と152はメモリ、160はバス・ブリッジ、200はノード、211と212はプロセッサ、221と222はタグ部、230はシステム・コントローラ、240はクロスバ・スイッチ、251と252はメモリ、260はバス・ブリッジ、300はノード、400もノードをそれぞれ示している。

【0011】ノード100、200、300、400は

同じ構成を持っているので、主としてノード100について説明する。バス・ブリッジ160の先には入出力装置が接続されている。プロセッサ111は内部にキャッシュ（図示せず）を有しており、キャッシュはデータ部とタグ部とから構成されている。プロセッサ112も内部にキャッシュを有している。タグ部121の内容はプロセッサ111のキャッシュのタグ部の内容と実質的に同じであり、タグ部122の内容はプロセッサ112のキャッシュのタグ部の内容と実質的に同じである。

【0012】システム・コントローラ130は、アドレスを一括制御するものである。クロスバ・スイッチ140は、システム・コントローラ130の指示に従って、データの流れを制御するものである。例えば、バス・ブリッジ160のデータ・バスをシステム・バスに含まれるデータ・バスに接続したり、プロセッサ111のデータ・バスをメモリ151のデータ・バスに接続したりする。メモリ151は主記憶の一部を構成するものである。メモリ152も同様である。例えば、メモリ151には主記憶アドレスのビット6～8が000であるデータが格納され、メモリ152には主記憶アドレスのビット6～8が001のデータが格納される。また、ノード200のメモリ251には主記憶アドレスのビット6～8が010のデータが格納され、ノード200のメモリ252には主記憶アドレスのビット6～8が011のデータが格納される。

【0013】タグ部121はN個（Nは2、3、…）のタグ情報を記憶することが出来る。タグ情報は、アドレスとキャッシュの状態を持つ。キャッシュの状態種別としては、M、O、S、Iがある。MはExclusive & Potentially Modifyを示し、OはShared-Modifyを示し、SはShare-Cleanを示し、IはInvalidateを示す。

【0014】タグ部121の中に存在する或るタグ情報のキャッシュの状態がMであると言うことは、当該タグ情報に対応する主記憶上のメモリ空間がプロセッサ111によって占有され、且つ当該タグ情報に対応するプロセッサ111のキャッシュのデータ・ブロックが書き換えられている可能性があることを示す。タグ部121の中に存在する或るタグ情報のキャッシュの状態がOであると言うことは、当該タグ情報に対応する主記憶上のメモリ空間がプロセッサ111を含む複数のプロセッサによって共有され、且つ当該タグ情報に対応するプロセッサ111のキャッシュのデータ・ブロックが書き換えられている可能性があることを示す。

【0015】タグ部121の中に存在する或るタグ情報のキャッシュの状態がSであると言うことは、当該タグ情報に対応する主記憶上のメモリ空間がプロセッサ111を含む複数のプロセッサによって共有され、且つ当該タグ情報に対応するプロセッサ111のキャッシュのデータ・ブロックが書き換えたものを共有している場合も

あることを示す。タグ部 121 の中に存在する或るタグ情報のキャッシュの状態が I であると言うことは、当該タグ情報に対応するプロセッサ 111 のキャッシュのデータ・ブロックが無効であることを示す。

【0016】ノード 100, 200, 300, 400 は、システム・バスによって接続されている。システム・バスは、アドレス・バス、データ・バス、リブライ・バス、スヌープ (snoop) バスから構成されている。各ノードのシステム・コントローラはアドレス・バス、リブライ・バス、スヌープ・バスに接続され、クロスバ・スイッチはデータ・バスに接続されている。

【0017】アドレス・バスには、以下に示すような複数の情報が入る。

○アドレス

○タイプ (アクセスの種類を示す。RDS, RDSA, ... など)

○マスタ ID (アクセス発生元。例えば CPU の ID など)

○クラス (処理の優先順位)

【0018】スヌープ・バスはシステム・コントローラから 4 本出ている。1 個のプロセッサあたり 2 本で、4 つのキャッシュの状態を示す。図示のように、4 個のシステム・コントローラが存在し、各システム・コントローラの配下に 2 個のプロセッサが存在する場合には、スヌープ・バスは 16 本の信号線で構成される。

【0019】リブライ・バスは複数本の信号線で構成される。以下に構成要素を示す。

○マスタ ID (どのマスタ ID のアクセスに介するリブライカを示す。)

○クラス (どのクラスに対するリブライカを示す。)

○タイプ (リブライの種類を示す。RBS, SWB, PSACK など)

RBS とは、Read Block Shared の略であり、メモリ・リードに対するリブライである。SWB とは、Slave Write Block の略であり、メモリ・ライトでデータをスレーブに送るリブライである。PSACK とは、インバリデート終了をマスタのシステム・コントローラに送るリブライである。

【0020】プロセッサが出すメモリ・アクセス要求には、以下に示すような 4 種類のリード要求と、2 種類のライト要求とが存在する。

RDS Read To Share

RDSA Read To Share Always

RDD Read To Discard

RDO Read To Own

WRB Write Back

WRI Write Invalidate

【0021】図 3 はシステム・コントローラの構成例を示す図である。同図において、131 はプロセッサ・バス制御部、132 はアドレス・バス制御部、133 はス

ヌープ・バス制御部、134 はクロスバ制御部、135 はメモリ制御部をそれぞれ示している。

【0022】プロセッサ・バスとは、プロセッサに接続されているアドレス・バスのことである。システム・コントローラ 130 は、プロセッサ・バス制御部 131 やアドレス・バス制御部 132、スヌープ・バス制御部 133、クロスバ制御部 134、メモリ制御部 135 を有している。他のシステム・コントローラも同様な構成を有している。

10 【0023】プロセッサ・バス制御部 131 は、プロセッサ・バスからデータを取り込んだり、プロセッサ・バスにデータを出力したりする。アドレス・バス制御部 132 は、システム・バスに含まれるアドレス・バスにデータを出力したり、これからデータを取り込んだりする。プロセッサ・バス制御部 131 とアドレス・バス制御部 132 の間には、データを遣り取りするための信号線が設けられている。

20 【0024】スヌープ・バス制御部 133 は、スヌープ・バスにデータを出力したり、スヌープ・バスからデータを取り込んだりすると共に、タグ部 121 の制御を行う。なお、図にはタグ部 121 しか示されていないが、タグ部 122 (図 2 を参照) もスヌープ・バス制御部 133 によって制御される。クロスバ制御部 134 は、クロスバ・スイッチやリブライ・バス、データ・バスの制御を行う。プロセッサ・バス制御部 131 とクロスバ制御部 134 の間にはデータを遣り取りするための信号線が設けられ、アドレス・バス制御部 132 とクロスバ制御部 134 の間にもデータを遣り取りするための信号線が設けられている。

30 【0025】メモリ制御部 135 は、メモリの制御を行う。クロスバ制御部 134 とメモリ制御部 135 の間の信号線はデータの転送に使用される。また、スヌープ・バス制御部 133 からメモリ制御部 135 には制御信号が送られる。

【0026】図 4 は本発明の動作を説明する図である。同図において、311 と 312 はプロセッサ、321 と 322 はタグ部、330 はシステム・コントローラ、340 はクロスバ・スイッチ、360 はバス・ブリッジをそれぞれ示す。～ は処理の順序を示す。なお、図 2 と同一符号は同一物を示す。また、ノード 200, 300 の中にはメモリ 151, 152 に相当するものが存在するが、図示されていない。

50 【0027】図 4 に示すように、プロセッサ 111 が或るメモリ空間 x に所有権付きのリード (キャッシュを書き換え可能な状態にするリード) をシステム・コントローラ 130 に対して要求したと仮定する。なお、所有権付きリードとは、前述の RDO のことである。システム・コントローラ 130 は、当該メモリ・リード要求をアドレス・バスを介してブロード・キャストする。なお、ブロード・キャストとは、計算機システムが複数のシス

テム・コントローラを持つ場合、一つのプロセッサが出してきたメモリ・アクセス要求を他のシステム・コントローラの全てに見せることを意味している。

【0028】システム・コントローラ130は、メモリ・リード要求をブロードキャストした後に、タグ部121, 122を参照して当該メモリ・リードの対象となるメモリ空間xに対応するキャッシュの状態を求め、求めたキャッシュの状態をスヌープ・バス上に出力する。なお、該当するタグ情報がタグ部に存在しない場合は、インバリッドがスヌープ・バス上に出力される。

【0029】システム・コントローラ230は、ブロード・キャストされたメモリ・アクセス要求を受信すると、タグ部221, 222を参照してメモリ空間xに対応するキャッシュの状態を求め、求めたキャッシュの状態をスヌープ・バス上に出力する。同様に、システム・コントローラ330は、ブロード・キャストされたメモリ・アクセス要求を受信すると、タグ部321, 322を参照してメモリ空間xに対応するキャッシュの状態を求め、求めたキャッシュの状態をスヌープ・バス上に出力する。

【0030】各システム・コントローラはスヌープ・バス上に出力されたキャッシュの状態を認識し、即座に必要な処理を開始する。例えば、スヌープ・バス上に出力されたプロセッサ211のキャッシュの状態がShared-Modify（キャッシュに最新のデータが存在する状態）、スヌープ・バス上に出力されたプロセッサ311のキャッシュの状態がShare-Clean（メモリ又は他のキャッシュの写しを共有している状態）、スヌープ・バス上に出力された他のキャッシュの状態がInvalid（無効）であったと仮定する。

【0031】システム・コントローラ230は、システム・コントローラ130によって所有権付きメモリ・リードが要求され且つ当該メモリ・リードの対象となるメモリ空間xに対応するタグ情報のキャッシュの状態がShared-Modifyであるので、プロセッサ211に対してコピーバック・インバリデート要求を発行する。

【0032】このコピーバック・インバリデート要求を受け取ると、プロセッサ211は、キャッシュのデータ部から該当するデータ・ブロックを読み出してシステム・コントローラ230に渡すと共に、キャッシュのタグ部に存在するメモリ空間xに対応するタグ情報のキャッシュの状態をInvalidにし、処理終了の応答をシステム・コントローラ230に返す。プロセッサ211からの処理終了の応答を受け取ると、システム・コントローラ230は、タグ部221の該当するタグ情報のキャッシュの状態をInvalidにし、渡されたデータ・ブロックをシステム・バスに含まれるデータ・バスに出力し、マスタのシステム・コントローラ130にリプライ・バスを經由して応答を返す。

【0033】システム・コントローラ330は、マスタのシステム・コントローラが所有権付きメモリ・リードを要求し且つプロセッサの311の該当するキャッシュの状態がShare-Cleanであるので、プロセッサ311に対してインバリッド要求を発行する。このインバリッド要求を受け取ると、プロセッサ311は、キャッシュのタグ部の中に存在するメモリ空間xに対応するタグ情報のキャッシュの状態をInvalidにし、処理終了の応答をシステム・コントローラ330に返す。

10 す。この応答を受け取ると、システム・コントローラ330は、タグ部321の中に存在する該当するタグ情報のキャッシュの状態をInvalidにし、マスタのシステム・コントローラ130にリプライ・バスを經由して処理終了の応答を返す。

【0034】一方、システム・コントローラ130は、各システム・コントローラ230, 330がキャッシュの状態をスヌープ・バス上に出力した所でスヌープ・バス上のキャッシュの状態を参照し、最新データがプロセッサ211のキャッシュにあると認識して、メモリ・リードを止めて、システム・コントローラ230からの応答およびシステム・コントローラ330からの応答を待ち、その後、プロセッサ111に対してリード・データを渡す。このリード・データを受け取ると、プロセッサ111は、リード・データをキャッシュのデータ部に格納すると共に、アドレスとExclusive&Potentially Modifyと言うキャッシュの状態を持つタグ情報をキャッシュのタグ部に書き込む。

【0035】また、上記の例に加えて、例えばプロセッサ212のキャッシュのタグ部の中に存在するメモリ空間xに対応するタグ情報のキャッシュの状態がInvalid（無効）ではなくてShare-Cleanの状態であると仮定すると、システム・コントローラ230はプロセッサ211に対してはコピーバック・インバリデートを発行し、プロセッサ212に対してはインバリデートを発行する。このような場合には、システム・コントローラ230はプロセッサ211, 212からの応答を待ち合わせ、両方からの応答があった段階でシステム・コントローラ130に処理終了を告げる応答を返す。このようにすることにより、マスタのシステム・コントローラに対する応答が一度で済む。

【0036】上述の説明はプロセッサ111から所有権付きのリードが発行された場合の動作を説明するものであるが、プロセッサ111から通常のリード要求（RDS/RDSA）が発行された場合の動作を以下に説明する。プロセッサ111が通常のリード要求を発行すると、システム・コントローラ130はシステム・バス上に当該リード要求をブロード・キャストし、システム・コントローラ130, 230, 330はタグ部の内容を確認して、スヌープ・バスにタグのステート（キャッシュの状態を示す）を出力する。

【0037】スヌープ・バス上のタグのステートが全てIであるとする、データはメモリから読み出される。何れのメモリからリードするかは、リード要求のアドレスによって決まる。

【0038】スヌープ・バス上に出力されたプロセッサ112のタグのステートがSであり、他のタグのステートがIであったとすると共に、リード要求で指定されるアドレスがメモリ251の空間であったと仮定する。この場合、リード要求で指定された最新データは、メモリ251およびプロセッサ112のキャッシュに存在する。他のノード200のメモリ251からノード100にデータを転送するよりも自ノード100内で終結する方が処理を早く終わらせることができ、システム・バスを使用しなくても済む。従って、プロセッサ112のキャッシュからデータを取ってくることを選択する。システム・コントローラ130は、プロセッサ112に対してコピーバックを発行し、この結果得られるデータをプロセッサ111にデータを転送する。リード要求で指定されるアドレスがメモリ151の空間であったと仮定する。この場合は、コピーバックよりメモリ・リードの方が速いので、システム・コントローラ130はメモリ151から該当するデータをリードし、リード・データをプロセッサ111に渡す。

【0039】スヌープ・バス上のプロセッサ212のタグのステートがMであり、他のプロセッサのタグのステートがIであったと仮定する。この場合は、最新データはプロセッサ212しか持っていないので、システム・コントローラ230がプロセッサ212にコピーバックを要求し、コピーバックの結果得られたデータをプロセッサ212からプロセッサ111に転送する。

【0040】次に、プロセッサがライト・インバリデイト(WRI)を発行した場合の動作について説明する。プロセッサ111からシステム・コントローラ130にライト・インバリデイト要求が出されると、システム・コントローラ130は当該要求をシステム・バス上にブロード・キャストし、各システム・コントローラ130、230、330はタグ部の内容を確認し、タグのステートをスヌープ・バス上に出す。このとき、各システム・コントローラは、スヌープ・バス上のタグのステートがI以外のS又はO又はMになっているプロセッサに対してインバリデイト要求を出し、インバリデイトが終了した段階でシステム・コントローラ130に応答を返す。システム・コントローラ130は全てのインバリデイトが終了した段階で、メモリ・ライトを行い、処理を終結させる。

【0041】図5はプロセッサ・バスからリード要求を受け取ったマスタのシステム・コントローラの動作を示す図である。同図において、PBはプロセッサ・バス、TWはシステム・バスをそれぞれ示す。

【0042】システム・コントローラは、プロセッサ・

バスからコヒーレント・リード(RDS, RDSA, RDDまたはRDO)を受け取ると、処理中のメモリ・アクセス要求のアドレスと今回のリード要求のアドレスが一致しているか否かを調べる。両者が一致している場合は、今回のリード要求をブロック・バッファに格納する。両者が一致していない場合は、今回のリード要求をシステム・バスに出力すると共に、今回のリード要求で指定されたメモリ空間が自己のノードの中に存在するか否かを調べる。自己のノードの中に存在する場合に

10 は、メモリからデータをリードしてリード・データをクロスバ・スイッチのバッファに入れるための処理を開始する。

【0043】リード要求をシステム・バスに出力した後、システム・コントローラはリード要求のアドレスによって定まるキャッシュの状態をスヌープ・バスに出力し、次いでスヌープ・バス上のキャッシュの状態を読み込む。そして、システム・コントローラは、読み込んだキャッシュの状態を参照して種々の動作を行う。

【0044】或る場合には、「PB SREQ CPB / I INV」を発行する。即ち、プロセッサ・バスを経由してプロセッサにコピーバックやコピーバック・イバリデイト、インバリデイト要求を送る。そして、プロセッサ・バスを経由してリプライやSACKが送られて来るのを待つ。「SACK」とは、インバリデイト終了のリプライである。

【0045】或る場合には、「TW Reply SACK」を待つ。即ち、システム・バスを経由してリプライやSACKが送られてくるのを待つ。ある場合には、「TW Reply RBU/S」を待つ。即ち、システム・バスを経由してRBS(リード・ブロック・シェア)又はRBU(リード・ブロック・アンシェア)のリプライが送られて来るのを待つ。RBUやRBSは、メモリ・リードに対するリプライである。或る場合にはメモリに対するリード処理をキャンセルし、或る場合にはメモリ・リードが終了するのを待つ。

【0046】待ち合わせ処理が終了した後、システム・コントローラは、RBU又はRBS又はOAK(オーナーシップ・アック)のリプライをマスタに送る。この場合のマスタとは、リード要求発行元のプロセッサである。また、必要に応じて「CRAB」をスレーブのプロセッサに送る。「CRAB」とは、コヒーレンシ・リード・アック・ブロックを意味している。

【0047】プロセッサ側にデータを送った後、システム・コントローラは「TW CEND」を発行する。

「TW CEND」は、システム・バス側の処理が終了したことを意味している。「TW CEND」が発行されると、CEND待ちになっているブロック・バッファの要求の待ち状態が解除され、次いでタグの更新が行われる。

50 【0048】図6はシステム・バスからリード要求を受



け取ったスレーブのシステム・コントローラの動作を示す図である。システム・コントローラは、システム・バスからコヒーレント・リード(RDS, RDSA, RD DまたはRDO)を受信すると、受信したリード要求で指定されたメモリ空間が自己のノードの中に存在するかどうかを調べる。自己のノードの中に存在する場合には、メモリからデータをリードしてリード・データをクロスバ・スイッチのバッファに入れるための処理を開始する。

【0049】リード要求を受信した後、システム・コントローラはリード要求のアドレスによって定まるキャッシュの状態をスヌープ・バスに出力し、次いでスヌープ・バス上のキャッシュの状態を読み込む。そして、システム・コントローラは、読み込んだキャッシュの状態を参照して種々の動作を行う。

【0050】或る場合には、「PB CPB/CPI +  $\alpha$  (INV)」を発行する。即ち、或るプロセッサに対してはコピーバック、コピーバック・インバリデート又はインバリデートを発行し、必要に応じて他のプロセッサに対してもコピーバック、コピーバック・インバリデート又はインバリデートを発行する。次いで、プロセッサ・バスを経由してリプライやSACKが送られて来るのを待つ。次いで、必要に応じて「CRAB」をスレーブのプロセッサに送り、システム・バスを経由してRBUやRBSのリプライをマスタのシステム・コントローラに送る。

【0051】或る場合には、プロセッサ・バスにインバリデートを発行し、プロセッサ・バス経由でSACKが送られて来るのを待ち、システム・バス経由でPSACKのリプライをマスタのシステム・コントローラに送る。

【0052】或る場合には、システム・コントローラは、メモリに対するリード処理をキャンセルする。メモリ・リードをキャンセルしない場合には、システム・コントローラは、メモリ・リードが終了したこと、インバリデートを発行した場合はSACKのリプライがあったことの両方の条件が満足した時に、システム・バスを経由してRBU/Sのリプライをマスタのシステム・コントローラに送る。

【0053】マスタのシステム・コントローラにリプライを送った後、システム・コントローラは、CEND待ちになっているブロッキング・バッファの要求の待ち状態を解除し、タグの更新を行う。

【0054】図7はプロセッサ・バスからWRB要求(ライトバック要求)を受け取ったマスタのシステム・コントローラの動作を示す図である。システム・コントローラは、プロセッサ・バスからWRB要求を受け取ると、処理中のメモリ・アクセス要求のアドレスと今回のWRB要求のアドレスが一致しているかどうかを調べる。両者が一致している場合は、今回のWRB要求をブロッ

キング・バッファに格納する。両者が一致していない場合は、今回のWRB要求をシステム・バスに出力する。

【0055】WRB要求をシステム・バスに出力した後、システム・コントローラはWRB要求のアドレスによって定まるキャッシュの状態をスヌープ・バスに出力し、次いでスヌープ・バス上のキャッシュの状態を読み込む。そして、システム・コントローラは、読み込んだキャッシュの状態を参照して種々の動作を行う。

【0056】或る場合には、「PB Sreplay WBCAN」を発行する。即ち、プロセッサ・バスを経由して「ライト・バックをキャンセルせよ」と言うリプライをWRB発行元のプロセッサに送る。

【0057】或る場合には、「PB Sreplay WAB」を発行する。即ち、プロセッサ・バスを経由して「ライトバックせよ」と言うリプライをWRB発行元のプロセッサに送る。次いで、WRBの対象となっているメモリ空間が自己のノードの中に存在するか、他のノードの中に存在するかどうかを調べる。自己のノードの中に存在する場合は、自己のメモリにWRBに付加されているデータを書き込む。他のノードの中に存在する場合には、「TW Reply SWB」を発行する。即ち、システム・バスを経由して「ライト・ブロックをメモリに書き込め」と言うリプライをスレーブのシステム・コントローラに送る。

【0058】メモリ・ライトなどの必要な処理を行った後、システム・コントローラは「TW CEND」を発行する。「TW CEND」が発行されると、CEND待ちになっているブロッキング・バッファの要求の待ち状態が解除され、次いでタグの更新が行われる。

【0059】図8はシステム・バスからWRB要求を受け取ったスレーブのシステム・コントローラの動作を示す図である。システム・コントローラは、システム・バスからWRB要求を受信すると、WRB要求のアドレスによって定まるキャッシュの状態をスヌープ・バスに出力し、次いでスヌープ・バス上のキャッシュの状態を読み込む。

【0060】次いで、システム・コントローラは、受信したWRB要求で指定されたメモリ空間が他ノードの中に存在するか、自ノードの中に存在するかどうかを調べる。自ノードの中に存在する場合には、マスタのシステム・コントローラから「TW Reply SWB」が送られて来るのを待ち、送られて来たならばメモリ・ライトを行い、「TW CEND」を発行する。「TW CEND」が発行されると、CEND待ちになっているブロッキング・バッファの要求の待ち状態は解除され、タグの更新が行われる。

【0061】図9はプロセッサ・バスからWRI要求(ライト・インバリデート要求)を受け取ったマスタのシステム・コントローラの動作を示す図である。システム・コントローラは、プロセッサ・バスからWRI要求

を受け取ると、処理中のメモリ・アクセス要求のアドレスと今回のWR I 要求のアドレスが一致しているか否かを調べる。両者が一致している場合は、今回のWR I 要求をブロッキング・バッファに格納する。両者が一致していない場合は、今回のWR I 要求をシステム・バスに出力する。

【0062】WR I 要求をシステム・バスに出力した後、システム・コントローラはWR I 要求のアドレスによって定まるキャッシュの状態をスヌープ・バスに出力し、次いでスヌープ・バス上のキャッシュの状態を読み込む。そして、システム・コントローラは、読み込んだキャッシュの状態を参照して種々の動作を行う。

【0063】或る場合には、「PB SREQ INV」を発行する。即ち、プロセッサ・バスを経由して「インバリデートせよ」と言う要求をキャッシュの状態がS、O又はMのプロセッサに送る。次いで、プロセッサ・バス経由でSACKが送られて来るのを待つ。或る場合には、システム・バス経由でSACKが送られて来るのを待つ。

【0064】或る場合には、「PB Sreply WAB」を発行する。次いで、WR I の対象となっているメモリ空間が自己のノードの中に存在するか、他のノードの中に存在するかを調べる。自己のノードの中に存在する場合は、自己のメモリにWR I に付加されているデータを書き込む。他のノードの中に存在する場合には、「TW Reply SWB」を発行する。すなわち、システム・バスを経由して「ライト・ブロックをメモリに書き込め」と言うリプライをスレーブのシステム・コントローラに送る。この処理は、待ち合わせが終了した後で行われる。

【0065】待ち合わせやメモリ・ライトが終了した後、システム・コントローラは「TWCEND」を発行する。「TW CEND」が発行されると、CEND待ちになっているブロッキング・バッファの要求の待ち状態が解除され、タグの更新が行われる。

【0066】図10はシステム・バスからWR I 要求を受け取ったスレーブのシステム・コントローラの動作を示す図である。システム・コントローラは、システム・バスからWR I 要求を受信すると、WR I 要求のアドレスによって定まるキャッシュの状態をスヌープ・バスに出力し、次いでスヌープ・バス上のキャッシュの状態を読み込む。そして、システム・コントローラは、読み込んだキャッシュの状態を参照して種々の動作を行う。

【0067】或る場合には、プロセッサ・バスにインバリデートを発行し、プロセッサ・バス経由でSACKが送られて来るのを待ち、送られて来たならばシステム・バス経由でSACKをマスタのシステム・コントローラに送る。或る場合には、WR I 要求で指定されるメモリ空間が自己のノードに存在するか否かを調べ、自己のノードに存在する場合にはシステム・バス経由でSWBが

送られて来るのを待ち、送られて来たならばメモリ・ライトを行い、「TW CEND」を発行する。「TW CEND」が発行されると、CEND待ちになっているブロッキング・バッファの要求の待ち状態は解除され、タグの更新が行われる。

【0068】

【発明の効果】以上の説明から明らかなように、本発明によれば、キャッシュのコヒーレンシーを保つための、システム・コントローラ間のバス・トランザクションを少なく出来る。また、プロセッサのキャッシュに対する処理の開始も早くなるため、システム全体の性能の向上が期待できる。

【図面の簡単な説明】

【図1】本発明の原理説明図である。

【図2】本発明のマルチプロセッサの構成例を示す図である。

【図3】本発明のシステム・コントローラの構成例を示す図である。

【図4】本発明の動作を説明する図である。

【図5】プロセッサ・バスからリード要求を受け取ったマスタのシステム・コントローラの動作を示す図である。

【図6】システム・バスからリード要求を受け取ったスレーブのシステム・コントローラの動作を示す図である。

【図7】プロセッサ・バスからWRB要求を受け取ったマスタのシステム・コントローラの動作を示す図である。

【図8】システム・バスからWRB要求を受け取ったスレーブのシステム・コントローラの動作を示す図である。

【図9】プロセッサ・バスからWR I 要求を受け取ったマスタのシステム・コントローラの動作を示す図である。

【図10】システム・バスからWR I 要求を受け取ったスレーブのシステム・コントローラの動作を示す図である。

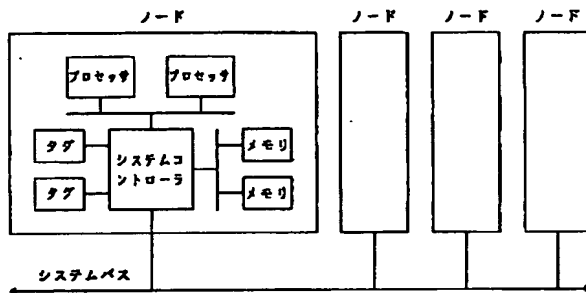
【符号の説明】

- 100 ノード
- 111 プロセッサ
- 112 プロセッサ
- 121 タグ部
- 122 タグ部
- 130 システム・コントローラ
- 140 クロスバ・スイッチ
- 151 メモリ
- 152 メモリ
- 160 バス・ブリッジ
- 200 ノード
- 211 プロセッサ

- 212 プロセッサ
- 221 タグ部
- 222 タグ部
- 230 システム・コントローラ
- 240 クロスバ・スイッチ
- 251 メモリ
- 252 メモリ
- 260 バス・ブリッジ

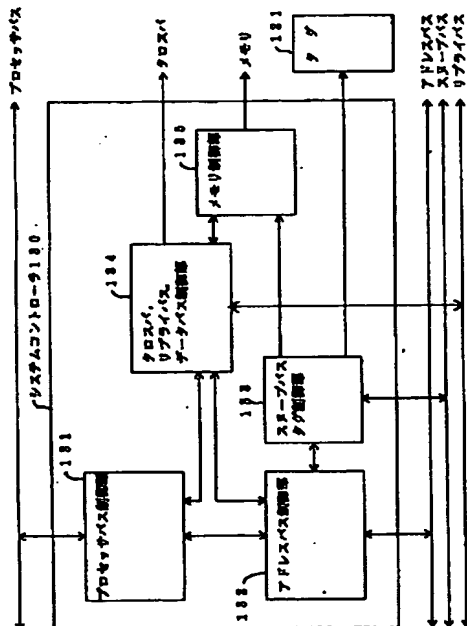
【図1】

本発明の原理説明図



【図3】

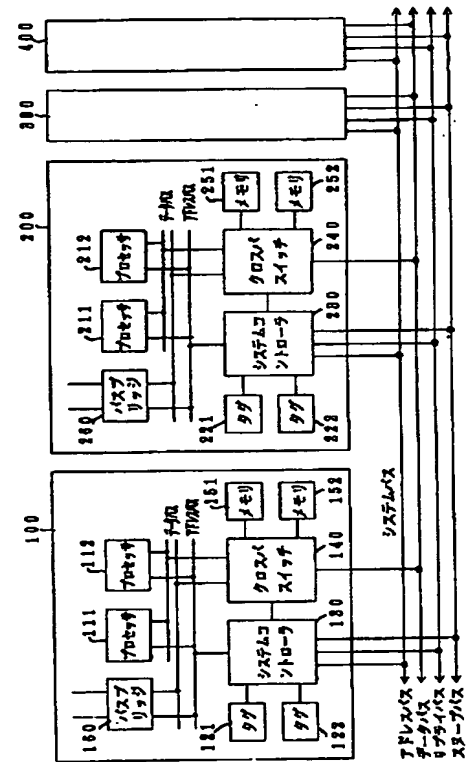
システム・コントローラの構成例



- 300 ノード
- 311 プロセッサ
- 312 プロセッサ
- 321 タグ部
- 322 タグ部
- 330 システム・コントローラ
- 340 クロスバ・スイッチ
- 360 バス・ブリッジ

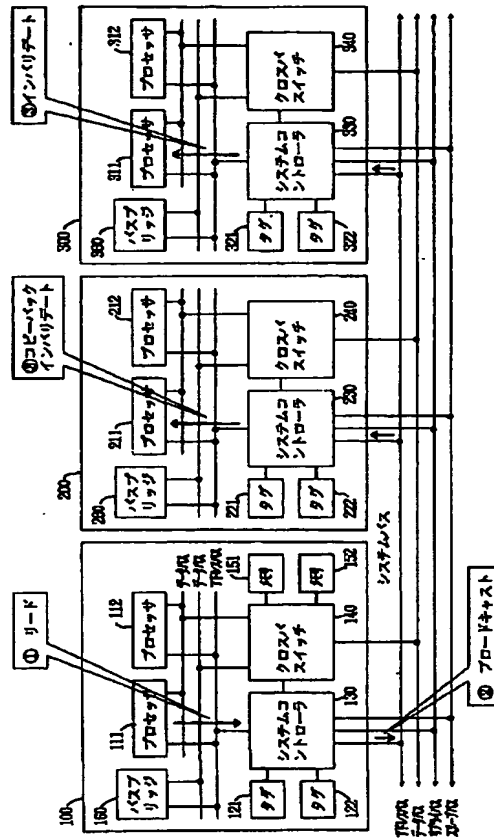
【図2】

本発明のマルチプロセッサの構成例



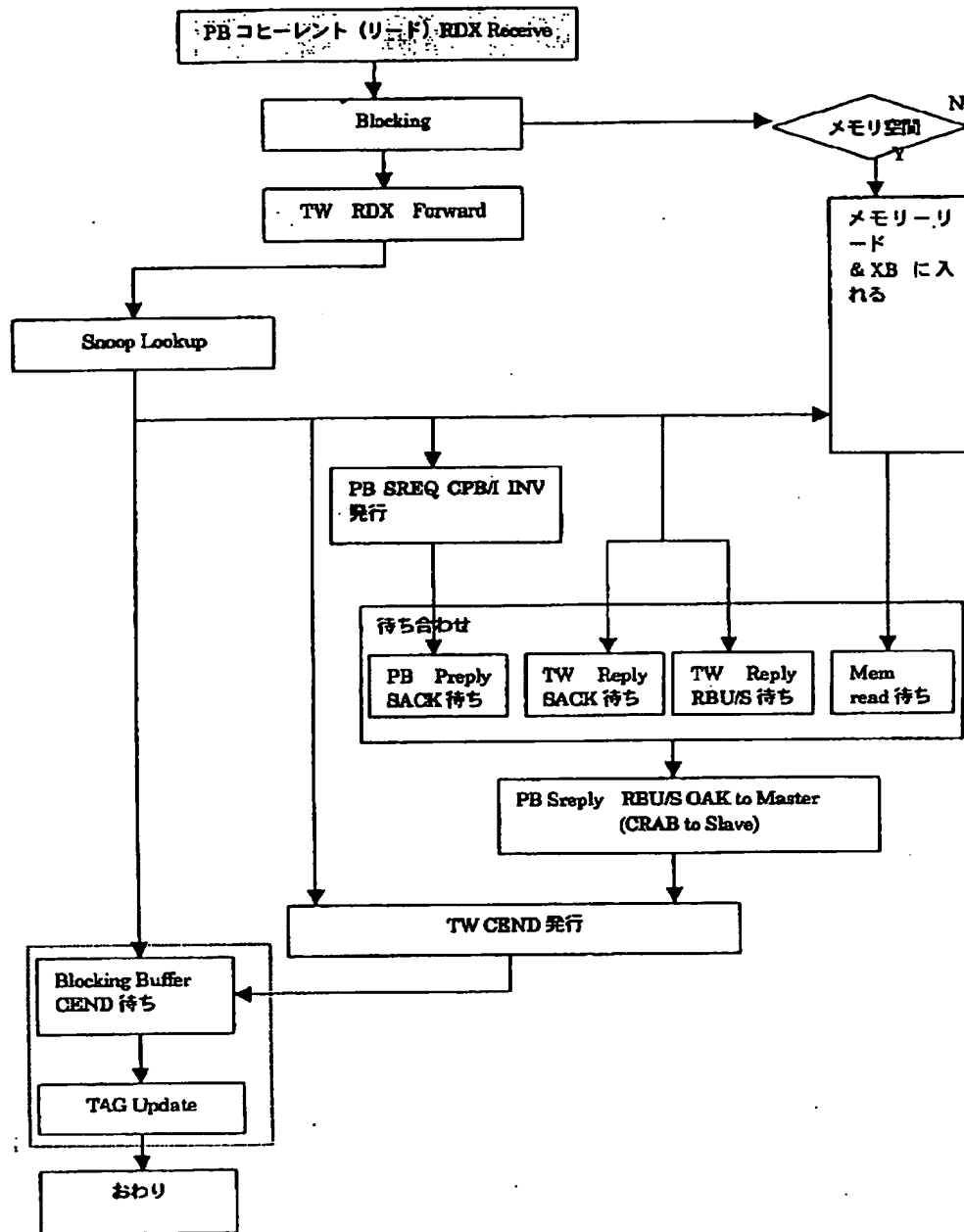
【図4】

本発明の動作を説明するための図



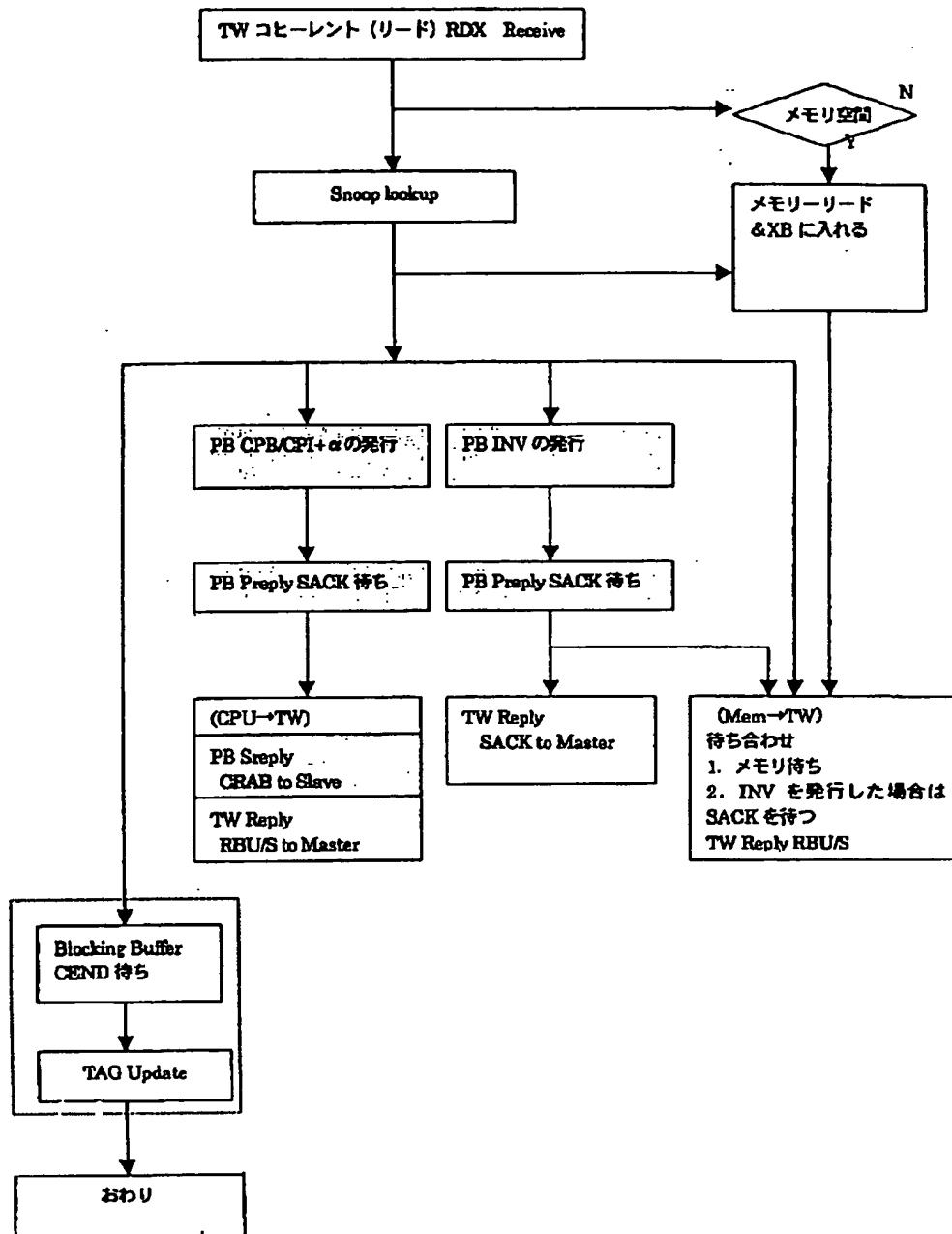
【図5】

プロセッサ・バスからリード要求を受け取ったマスタのシステム・コントローラの動作



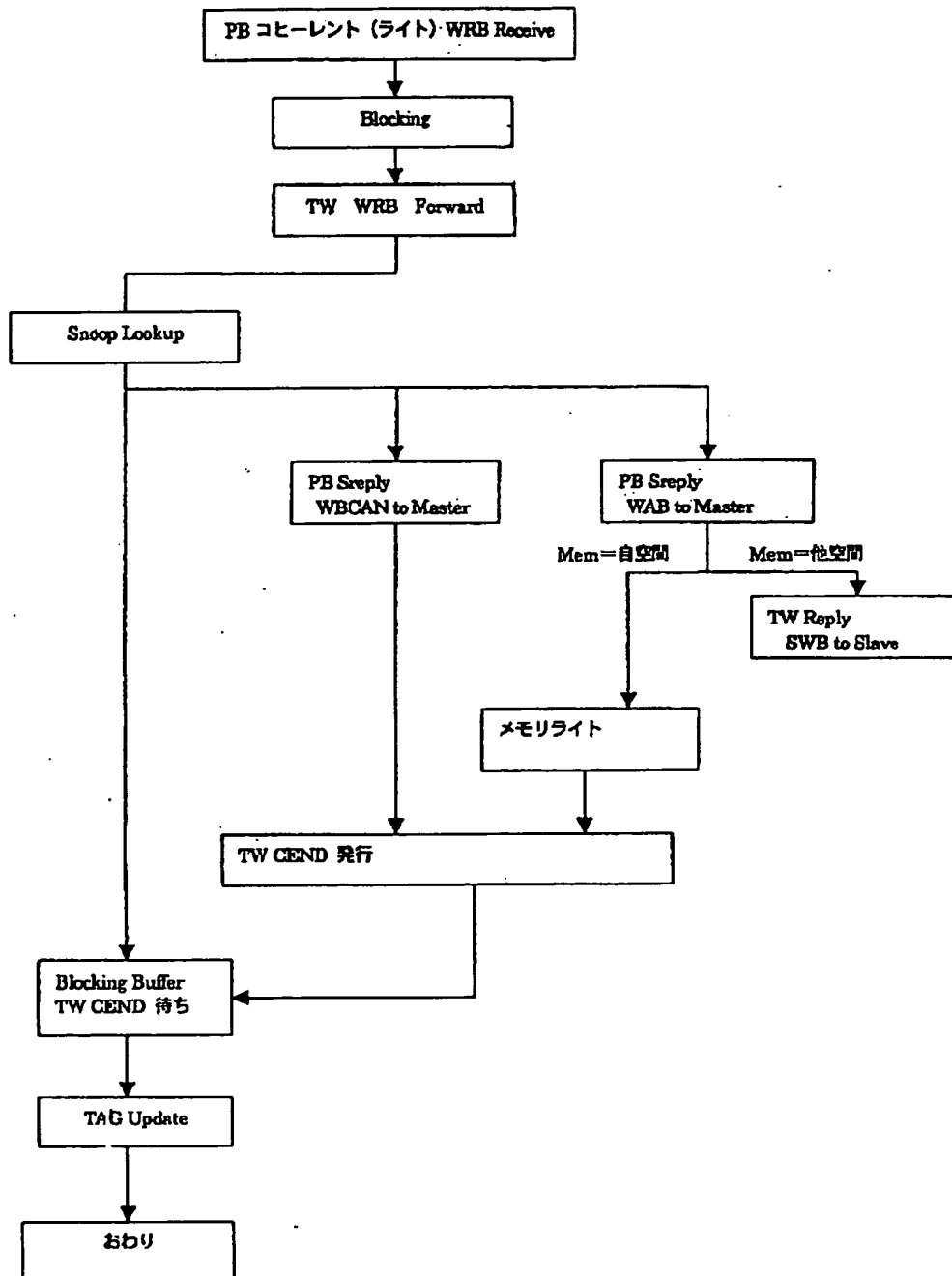
【図 6】

システム・バスからリード要求を受け取ったスレーブのシステム・コントローラの動作



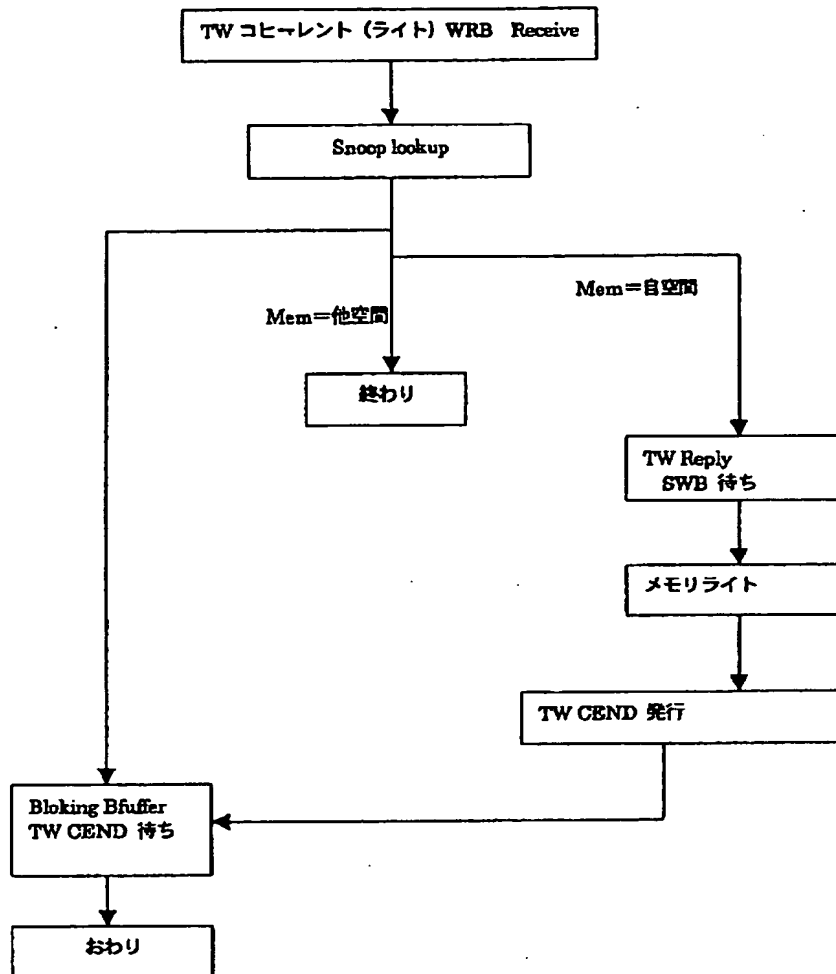
【図 7】

プロセッサ・バスから WRB 要求を受け取ったマスタのシステム・コントローラの動作



【図8】

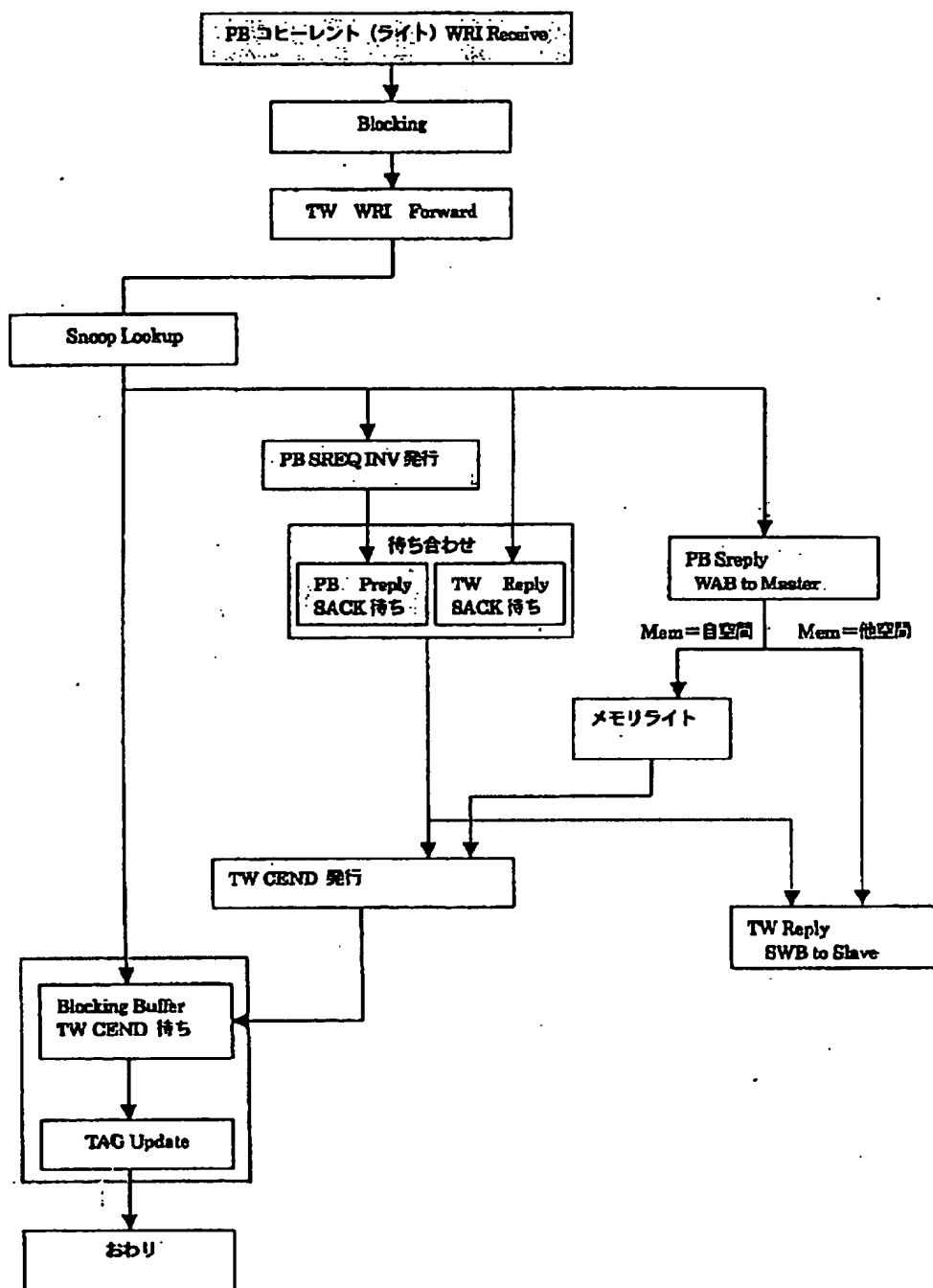
システム・バスからWRB要求を受け取ったスレーブのシステム・コントローラの動作





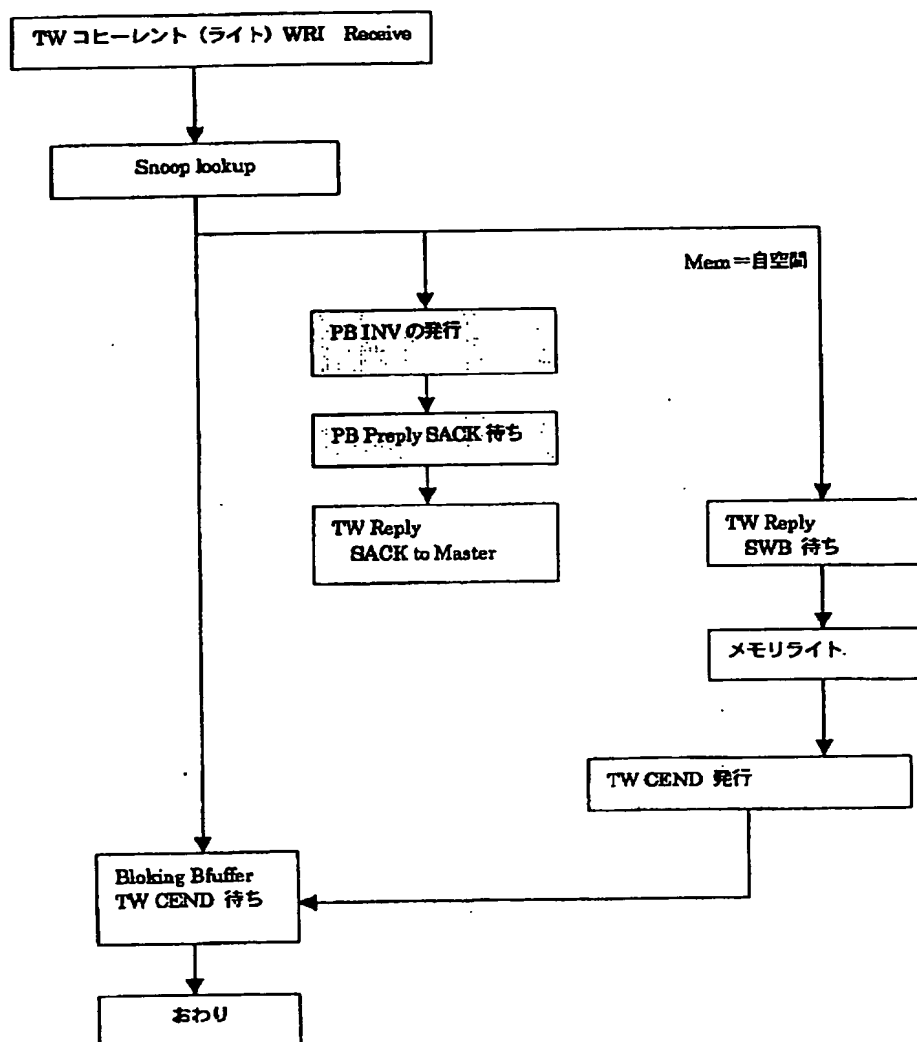
【図9】

プロセッサ・バスからWR I 要求を受け取ったマスタのシステム・コントローラの動作



【図10】

システム・バスからWR1要求を受け取ったスレーブのシステム・コントローラの動作



フロントページの続き

(72)発明者 佐藤 正美  
石川県河北郡宇ノ気町宇野気ヌ98番地の  
2 株式会社ピーエフユー内

Fターム(参考) 5B045 BB12 DD12 DD13